

3 Hours Duration

NOTES:

1. If doubts exist as to the interpretation of any question the candidate is urged to submit with the answer paper a clear statement of any assumption made.
2. Provide justifications for your answers. Show all your work.
3. CLOSED BOOK. Candidates may use one of the two pocket calculators; the Casio approved model or sharp approved model. No other aids.
4. The candidate has to answer five questions (each question has multiple parts).
Answer ANY ONE of questions 1 and 2, ANY ONE of questions 3 and 4, ANY THREE of questions 5, 6, 7, and 8.
5. Total Marks = 100.
6. This exam has got 6 pages (including this page).

1. [20 marks]

(a). Consider a system in which four processes P1, P2, and P3 execute concurrently. Each process has a unique integer number associated with it. I1, I2, and I3 are the numbers associated with P1, P2, and P3 respectively. The system contains a shared data record, which may be concurrently accessed by these processes. A **monitor** is used to control access to the shared data. A number of processes are allowed to access the shared data simultaneously as long as the sum of the numbers for the processes concurrently accessing the data is **less than a given integer constant I**. A requesting process is blocked if allowing the process to access the data will violate this condition. A blocked process is allowed to proceed when one or more of the processes exit the monitor and allowing the blocked process to continue will not violate this condition. For example, If P_i and P_j are currently accessing the shared data and P_k wants to concurrently access the data, the access is allowed if $I_i + I_j + I_k < I$. P_k is blocked otherwise. If P_i leaves the monitor after P_k is blocked, P_k is allowed to proceed if and only if $P_j + P_k < I$.

The monitor contains two procedures: `want_access` and `finish_access`. Whenever a process wants to access the record it calls the procedure `want_access`. The integer number associated with the process is passed as an argument to the procedure. If the calling process satisfies the condition described earlier it is allowed to proceed; it is **blocked** otherwise.

Whenever a process finishes its operation on the data record it calls the procedure `finish_access`. The calling process may pass any parameter to this procedure that you feel is necessary. If one or more processes is/are blocked when `finish_access` is called, the blocked process/processes is/are allowed to access the record after the completion of the `finish_access` procedure. The process/processes that are allowed access is/are selected in such a way that the condition on sum of the numbers associated with the processes concurrently accessing the data is satisfied.

The typical operations performed by a process P_j ($j = 1 \dots 4$) is given by the following algorithm.

Process P_j

repeat

1. Perform computation.
2. Call procedure `want_access` in the monitor. [Indicate the number I_j associated with the process through the argument of the procedure.]
{If the process is not blocked inside the monitor it means that the desired operation on the shared record can be performed by the process.}
3. Perform the desired operation on the data record.
4. Call procedure `finish_access` in the monitor.

until false

Write the algorithm (pseudo-code) for the monitor that will control access to the data record. The monitor must contain the two procedures `want_access` and `finish_access` (described above) that are called by the processes. You may incorporate other procedures/functions inside the monitor if necessary.

2 [20 marks]

(a) Given below is a solution to the critical section problem involving two concurrent processes P0 and P1. Identify as many **distinct problems** as you can in the design. If similar problems occur at multiple places identify them each time but explain it only once. Your list of errors should include defects (if any) that may not necessarily give rise to incorrect results but do indicate flaws in design. Justify your answer with the help of examples. Be as specific as you can when you describe the situations in which problems occur.

Algorithm

<u>Process P0</u>	<u>Process P1</u>
repeat	repeat
need [0] := true;	need [1] := true;
if need [1] then	if need [0] then
begin	begin
need [0] := false;	need[1]:=false;
while need[1] do no-op;	while need[0] do no-op;
need [0] := true;	need [1] := true;
end;	end;
Code for CS	Code for CS
need [0] := false;	need [1] := false;
until false;	until false;

Note: CS: Critical Section. no-op: no operation.

(b) Briefly discuss the following:

(i) Why and where condition variables are used.

(ii) Discuss how a semaphore can be used to satisfy the following requirements associated with the solution to the critical section problem: **mutual exclusion** and **freedom from starvation**.

- (iii) The difference between synchronous and asynchronous message passing.

Answer Question 3 or Question 4

3 [20 marks].

(a). Consider a demand paged virtual memory system in which a single program is currently running. The page map table is held in associative registers (associative memory). It takes E milliseconds to service a page fault if an empty frame is available or the replaced page is not modified, and M milliseconds if the replaced page is modified. Memory access time is A nanoseconds and F is the probability of a page fault.

Assume that for $R\%$ of the page faults a page replacement is necessary and the page to be replaced is modified. Derive an expression for the **effective memory access time** for the program.

(b). Consider a demand paged virtual memory system and the reference string:

51,52,53,54,52,51,55,56,52,51,52,53,57,56,53,52,51,52,53,56

With 3 frames allocated to the program determine the total number of page faults for the following cases.

- (i) when the LRU page replacement strategy is used.
- (ii) when the optimal page replacement strategy is used.

4 [20 marks].

(a) Describe how multiple partition-based memory management is performed on a computer system.

(b) Different strategies for mapping an incoming job to a memory partition can be used when the incoming job's memory requirement can be satisfied by multiple partitions. Describe three strategies for selecting a memory partition for the new job. Include the advantages/disadvantages of each strategy in your discussion.

(c) What are the shortcomings of multiple partition-based memory management? Describe a technique that can alleviate these shortcomings.

Answer any THREE of questions 5, 6, 7, and 8.

Question 5 [20 marks].

(a) Consider a **preemptive** short term scheduling strategy in which the priority of a process may change dynamically with time. (Larger priority numbers imply higher priority). At any point in time the highest priority process is run on the system. Ties are broken in favour of the process that entered the ready to run queue first. If a running process is preempted then its time of entry into the ready to run queue is the time at which the preemption was made.

98-COMP-A5 / SOFT-A5 OPERATING SYSTEMS

The priority of all processes is set to **P** when they enter the ready to run queue (upon arrival or after being preempted). When a process is waiting in the ready to run queue its priority changes at a rate **a**. That is,

Priority of a process in the ready to run queue = $P + at$
(where t is the time elapsed (in seconds) after the process entered the ready to run queue).

When a process is selected to run on the CPU its priority is set to **Q**. As it starts running on the CPU its priority changes at a rate **b**. That is,

Priority of the process running on the CPU = $Q + bt'$
(where t' is the time elapsed (in seconds) after the process started running on the CPU).

The parameters P , Q , a , and b can be set to give many different scheduling policies. Once chosen the values of these parameters become fixed and can not change.

Determine P , Q , a , and b that will produce the Last Come First Served policy. Under this policy whenever a process arrives on the system it preemptively captures the CPU. That is, if the CPU is free the process is allocated the CPU; if the CPU is busy then the executing process is preempted and the CPU is allocated to the process that just entered the ready to run queue. Whenever a process completes its CPU burst and the CPU is free the process (in the ready to run queue) that was preempted most recently is allocated the CPU.

(b) Using examples distinguish between a hard and a soft real time system. Discuss the goals of the scheduler deployed in each of these type of systems.

6 [20 marks].

(a) Briefly discuss any two strategies for managing free space on the disk. Include the advantages and disadvantages (if any) of the strategies in your discussion.

(b) Consider a moving head hard disk which consists of a single platter (surface) with 250 tracks on it. The tracks are numbered 0 to 249. The disk is currently serving a request at track 150 and has just finished a request at track 148. The queue of pending requests in FIFO order is:

96, 157, 101, 187, 104, 160, 112, 185, 140.

What is the total head movement (in number of tracks) needed to satisfy all these requests for the following disk scheduling algorithms?

(i) Shortest Seek Time First (SSTF) (ii) LOOK

[Assume that no further requests arrive on the system during the service of the above requests.]

(c) Consider a single user environment. Which of the strategies, SSTF or FCFS would you use?

7 [20 marks].

(a) Distinguish between deadlock and starvation (using examples).

(b) Consider a multiprogrammed system consisting of five resources of the same type. Four processes are run on the system. Each process can simultaneously hold up to two resources at any given point in time.

Once a device is acquired by a process it must be released by the process before it can be assigned to another process. Assume that each process requests and releases one resource at a time.

Show that a deadlock can never occur on the system and it is not necessary for the system to deploy any deadlock handling technique.

(c) Does an unsafe state always lead to a deadlock state? Justify your answer using an example.

8 [20 marks].

(a) Briefly discuss File Allocation Table (FAT).

(b) Why is processor scheduling considered to be a harder problem in a multiprocessor system in comparison to a single CPU system?

(c) Briefly discuss the strategy based on user groups used for file protection. Include the advantages/disadvantages of the strategy in your discussion.

(d) Give examples of applications that perform (i) sequential access to files (ii) random access to files.

(e) Briefly discuss the goals of *protection* in a multi-user computer system.